Introduction to EZ Pool

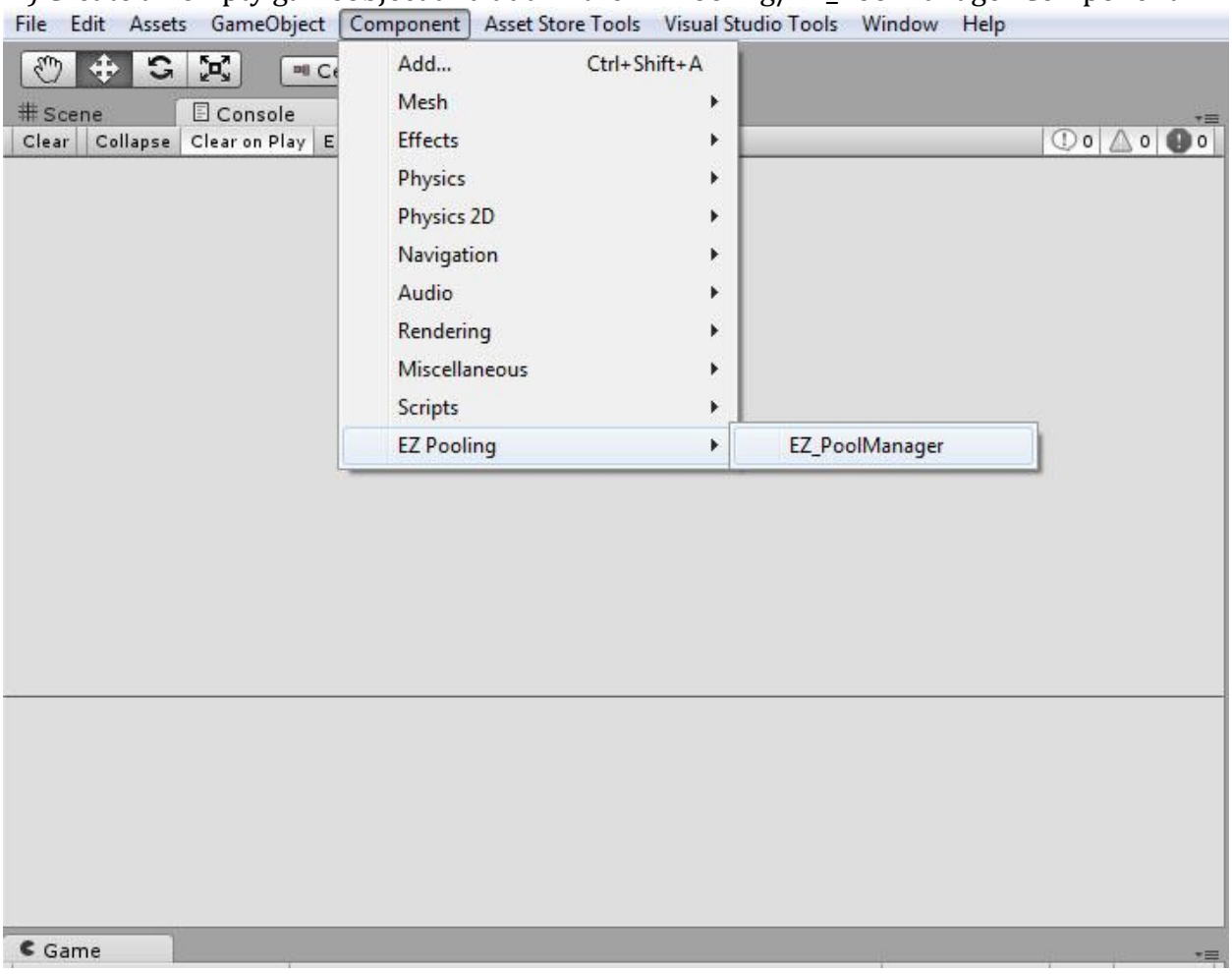
EZ Pooling is a simple asset that can make any game become more efficient and run faster.

EZ Pooling helps to keep track of the objects that are needed over and over again in the scene. By managing a pool of objects and activating / deactivating them via Spawn() and Despawn(), EZ Pooling bypasses the time expensive functions of calling Destroy and Instantiate.

EZ Pooling is easy to integrate to any project, just replace Instantiate(...) with Spawn(...) and Destroy() with Despawn()

Quick start guide

1) Create an empty gameobject and add in the EZ Pooling/EZ_PoolManager Component.



2) Set the options.

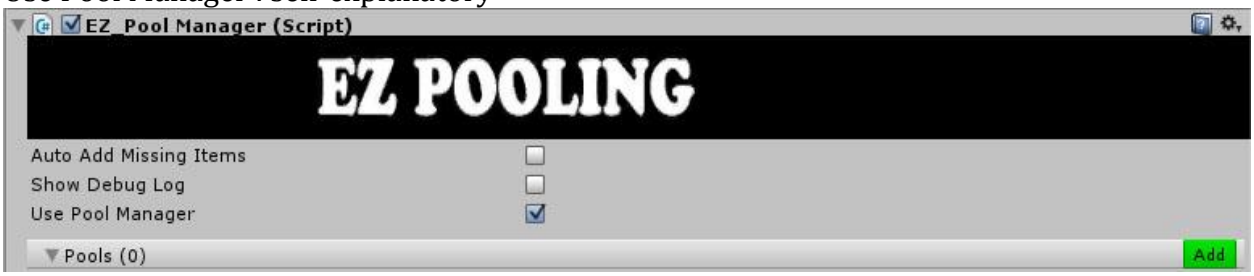
Auto-Add Missing Items : allow the pool manager to smart allocate new pools when required during run-time.

The default options given to the newly created pools can be tweaked at :

```
/// <summary>
        /// Method to create a new pool during run time. 'auto Add Missing Items' must be
enabled
        /// </summary>
        private static void CreateMissingPrefabPool(Transform missingTrans, string name)
        {
            var newPrefabPool = new EZ_PrefabPool();

            //Set the new pool options here
            newPrefabPool.parentTransform = parentTransform;
            newPrefabPool.poolCanGrow = true;

            Pools.Add(name, newPrefabPool);

            // for the Inspector only
            var newPrefabPoolOption = new EZ_PrefabPoolOption();
            newPrefabPoolOption.prefabTransform = missingTrans;
            newPrefabPoolOption.poolCanGrow = true;
            EZ_PoolManager.Instance.prefabPoolOptions.Add(newPrefabPoolOption);

            if (EZ_PoolManager.Instance.showDebugLog)
            {
                Debug.Log("EZ_PoolManager created Pool Item for missing item : " + name);
            }
        }
```

Show Debug Log : log events, warnings, errors

Use Pool Manager : self-explanatory



3) Click Add to add a pool

4) Assign the prefab that you want to pool.

5) The final thing is to replace the Instantiate and Destroy to Spawn and Despawn.

For each object that are in the pool, make sure that you assign a script that has the method OnSpawned() and OnDespawned() as Awake() or Start() won't be called by the Pool Manager. And move the initializing logic to OnSpawned().

Example

```
public class basic_object : MonoBehaviour
{
    void OnSpawned()
    {
        //this method will be called when an object is spawned by the pool manager
        if (rigidbody)
            rigidbody.velocity = Vector3.zero;
    }

    void OnDespawned()
    {
        //this method will be called when an object is despawned by the pool manager
    }
}
```

See examples in the project folder for more details.

Advanced Options.

Allow Pool to grow : allow pool to grow when there is demand for more objects in the scene

Cull Despawned : destroy excess despawned obj to free memory space

Allow Pool to recycle : allow the pool to reuse the oldest active game obj when the hard limit is already reached, or if the pool is not allowed to grow.

That's it for now, I hope this asset will make your game more awesome.

Support Contact : rudinesurya@gmail.com